

<! --Topicos Especiales y Avanzados-->

<! --Ing. Amy Diaz-->

Documentación de API {

<Por="Karen Lopez"/>

<Por="Junior Garcia"/>

}



Introducción {

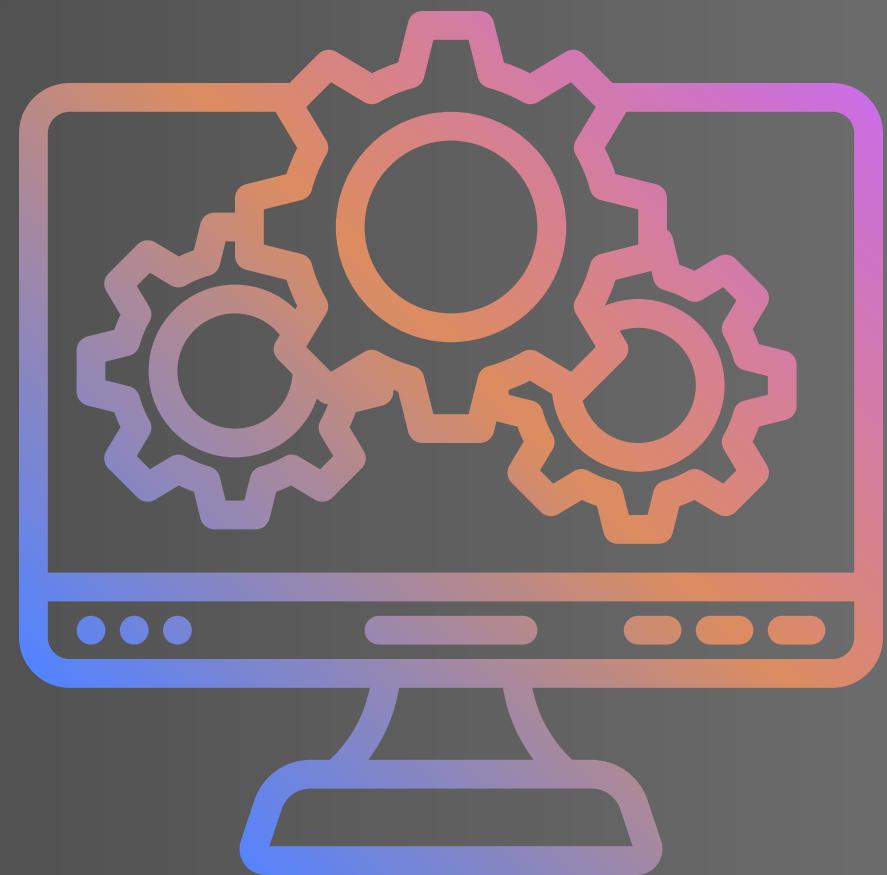
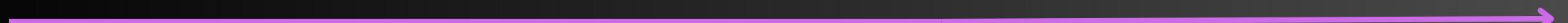
¿Que es un API?

Interfaz de
Programación de
Aplicaciones

}



Beneficios y ejemplos de uso APIs

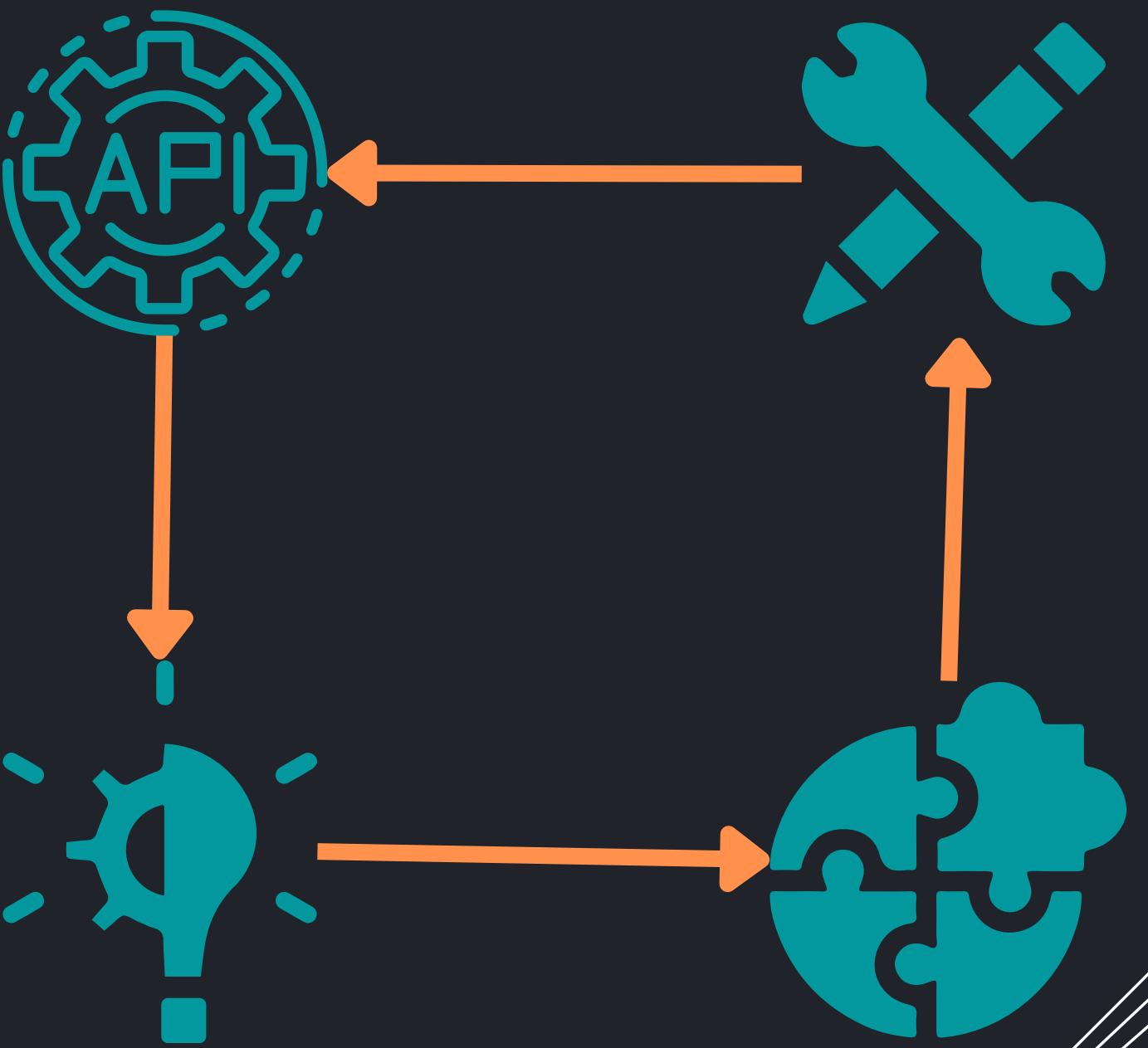


Beneficios de API

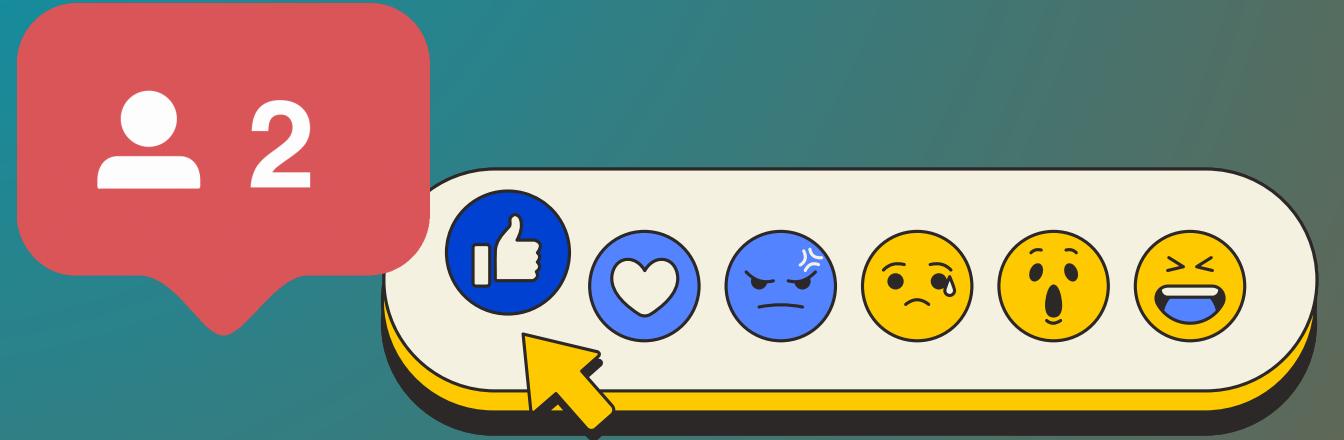
REST {

- Integración
- Innovación
- Ampliación
- Facilidad de mantenimiento

}



Ejemplos de uso cotidiano



APIs de redes sociales

APIs de pasarelas de pago

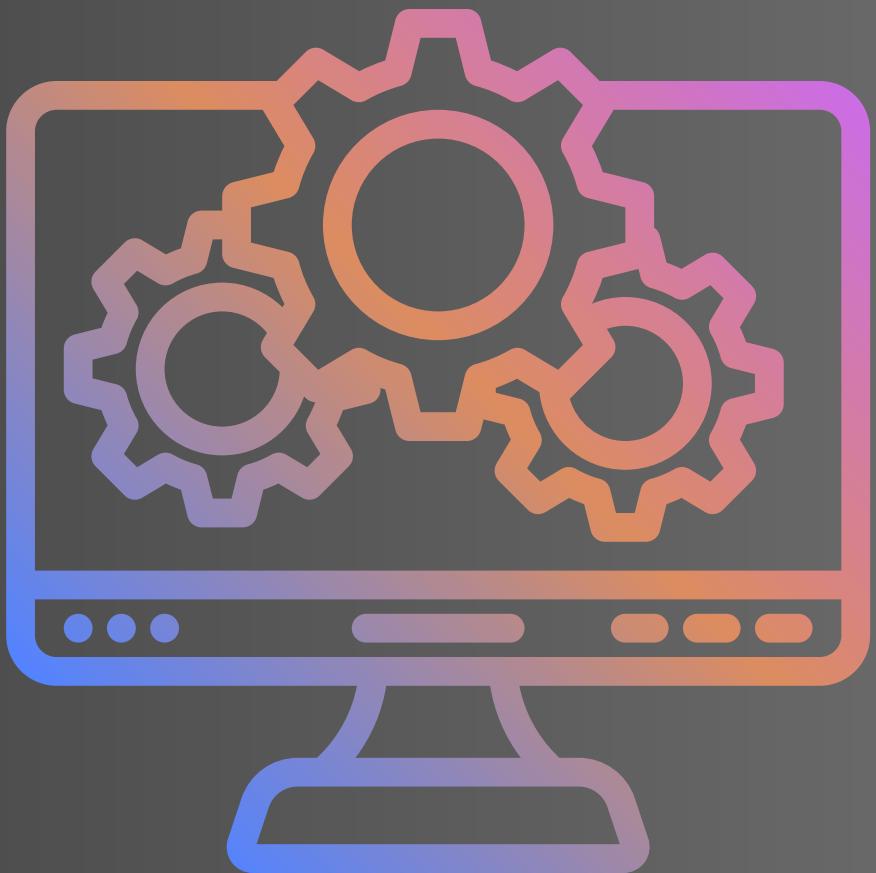


APIs de servicios en la nube

Ciclo de vida de las APIs



Tipos de Documentación de API

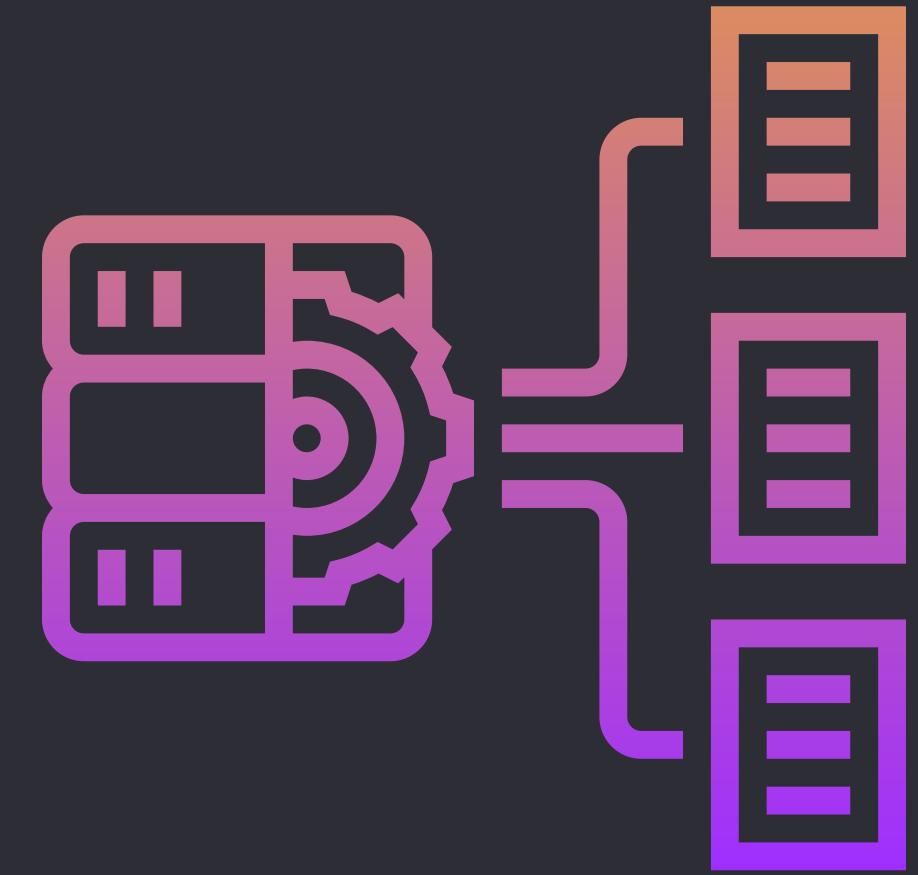
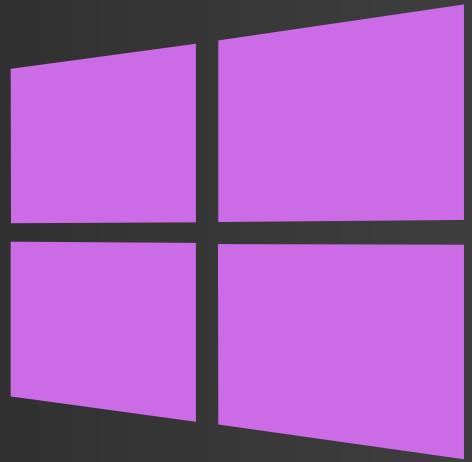


Documentación de referencia

Proporciona información técnica detallada sobre todos los endpoints, métodos, parámetros, tipos de datos, códigos de estado y cualquier otro aspecto técnico de la API.



Ejemplos de documentación de referencia

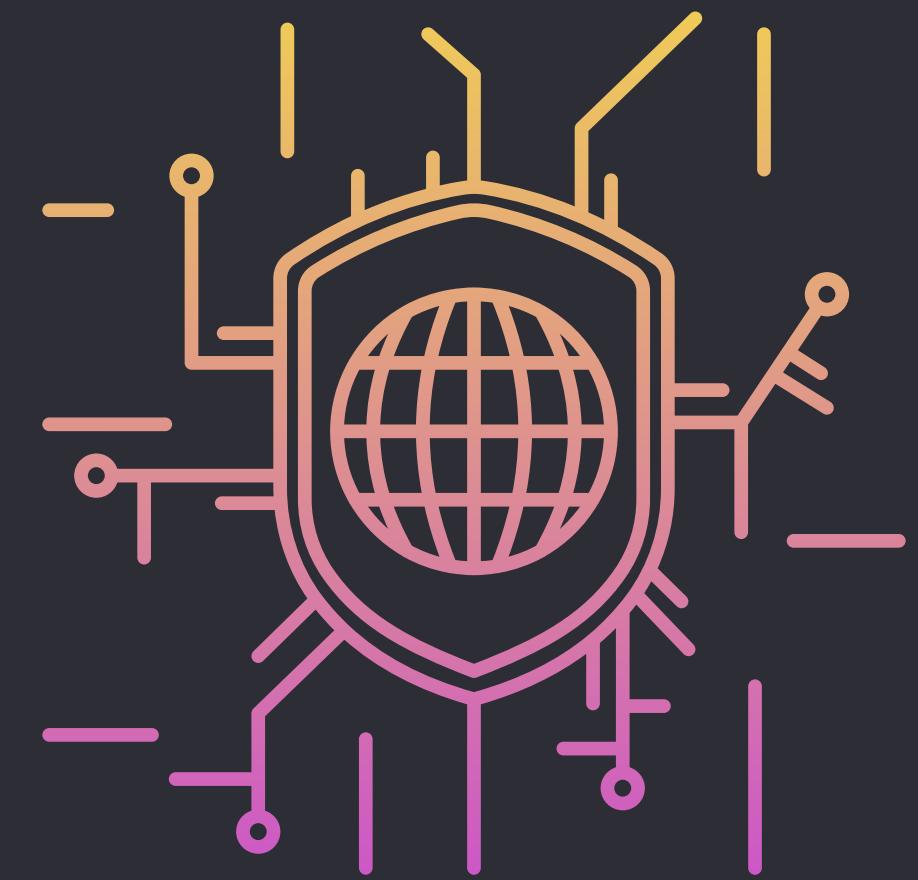
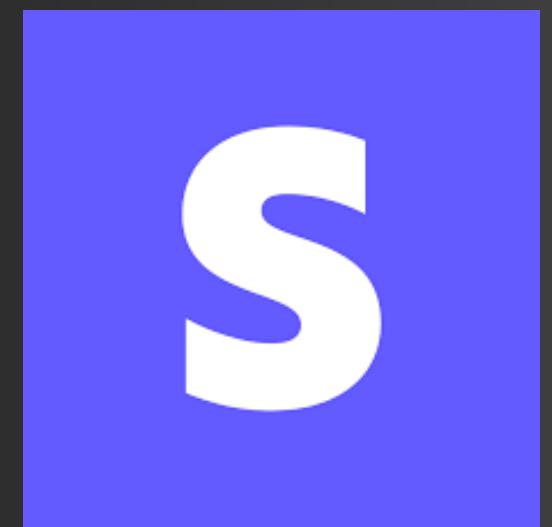


Documentación de uso

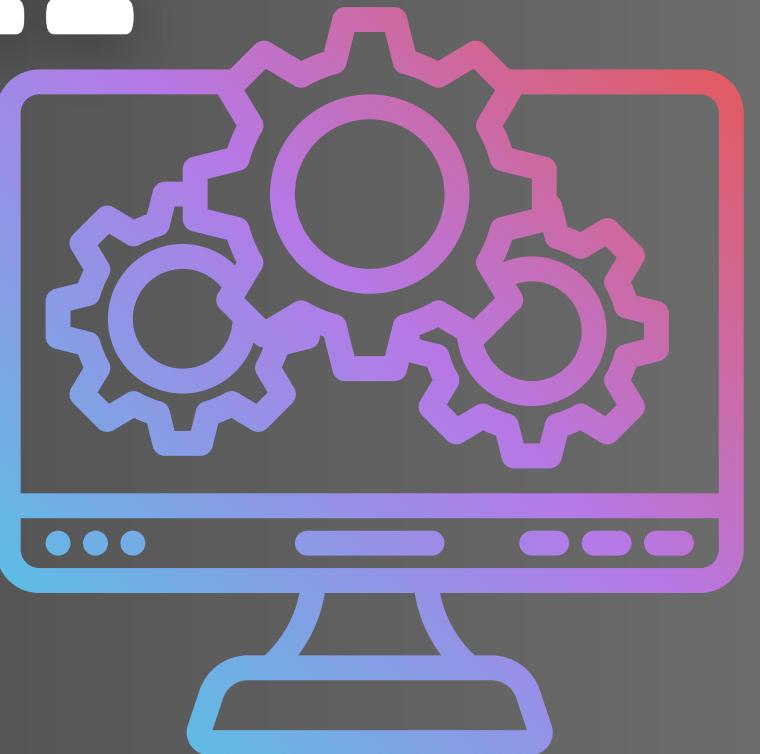
Se centra en cómo utilizar la API de una manera más general. Incluye ejemplos y casos de uso que ayudan a los desarrolladores a comprender cómo aprovechar al máximo la API



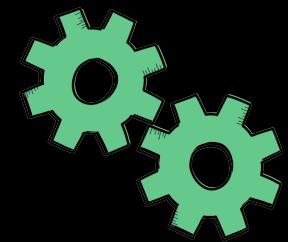
Ejemplos de documentación de uso



Importancia de una Documentación



Si tu API no tiene documentación, nadie
usará tu API en absoluto

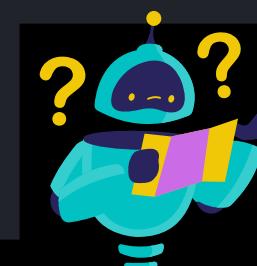


Una buena documentación ahorrará tiempo

en entender cómo funciona el API



Reducirá el número de preguntas
innecesarias que tendría que responder
el equipo de desarrollo



Errores de documentación

01 - No subirlos al repositorio

02 - No actualizar documentos/código

03 - No tener orden en el versionamiento

04 - No tener orden en las carpetas (arq, dev, qa)

05 - No seguir el estandar (plantillas)

Elementos clave de una buena documentación



Elementos clave de una buena documentación {

[Título API]

[Descripción]

`http://<SERVER_ADDRESS>/`

Método: GET/PUT/POST/DELETE

Authentication: basic authentication/API key/OAuth

Request Parameters

Header fields

Header Field	Value
Authorization	A valid access token

Parameters

Parameter	Type	Description	
id	string	User ID	REQUIRED

1 TITULO/INTRODUCCION

2 ENDPOINT

3 METODO

4 AUTENTICACION

5 PARAMETROS

6 JSON REQUEST BODY

JSON Request Body

Value	Type	Description	
serialNumber	string	<i>Serial Number</i>	REQUIRED

HTTP Response

HTTP Code	Description
200	Successful Response
401	Unauthorized

Error or Warning Response

invalid value for `field_name`

Unrecognized values for parameter paramname: value1, value2, value3

7 RESPONSE

8 EJEMPLO

Plantilla -> { }

Mejores prácticas en la Documentación





Mejores prácticas en la Documentación de API {



Planificar

Tener un plan de acción para elaborar la documentación

Actualizar la documentación

Asegurar que la documentación sea actualizada periódicamente



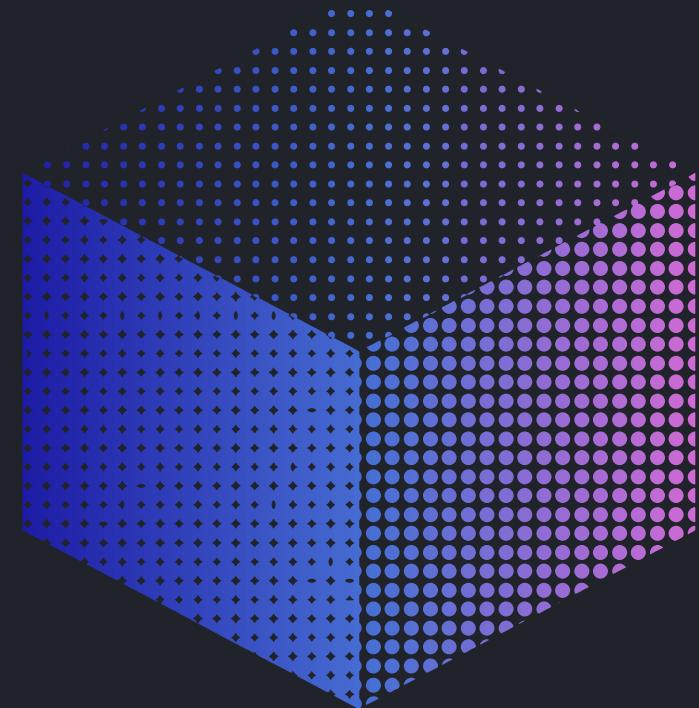
Responsable

Encargado de desarrollar y mantener la documentación



Pensar en la audiencia

Escribir la documentación pensando en la audiencia que la usará



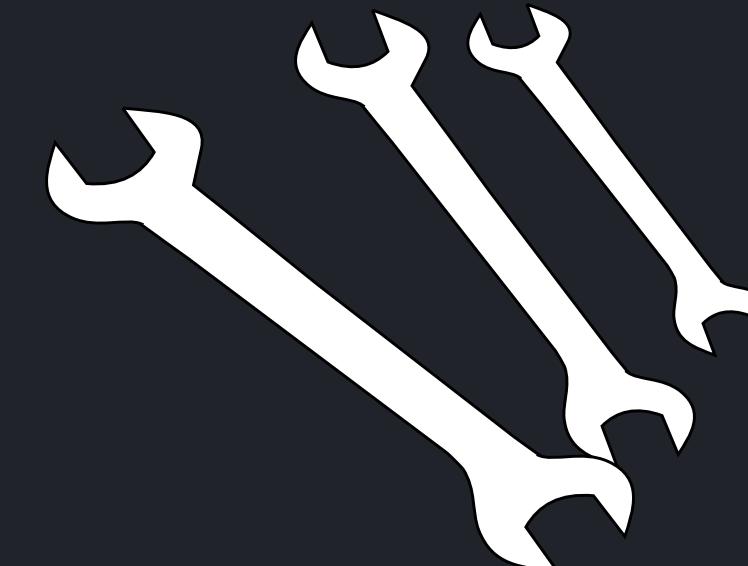
Consistencia y evitar jerga

Documentación
uniforme y con
la misma
terminología y
nomenclatura



Crear ejemplos

Incluir código
funcional y que
pueda ser testeado
de acuerdo
a la documentación
sugerida

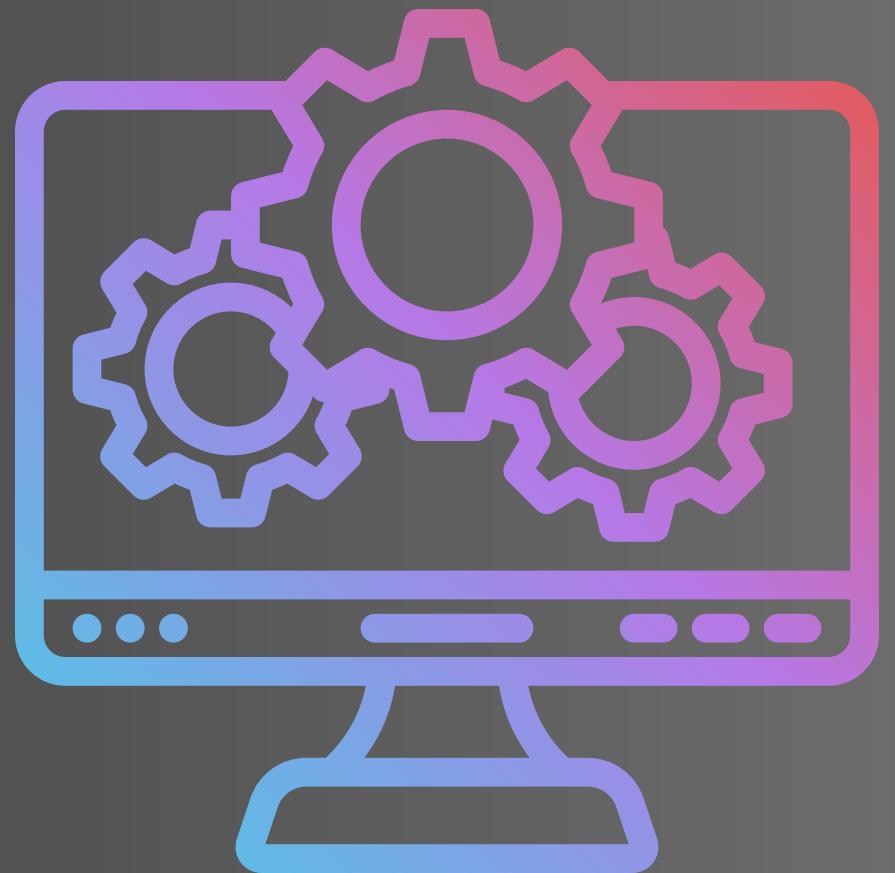


Usar herramientas

Permiten facilitar
el trabajo y ayudan
a crear la
documentación de
forma dinámica



Herramientas de Documentación de API



HERRAMIENTAS{

Swagger

The screenshot shows the Swaggerhub interface. On the left, there is a code editor window titled "swagger-hub... | registry-a... | 1.0.45" containing a Swagger JSON specification. The specification includes details like the swagger version (2.0), info (description, introduction, authentication instructions), and various API definitions with their methods (GET, POST, DELETE, PUT) and paths. On the right, there is a list of "APIs: Operations for APIs" with corresponding HTTP methods and descriptions. The interface has a dark theme with some UI elements in green and red.

swagger-hub... | registry-a... | 1.0.45

Editor Split UI

Read Only

```
1: swagger: '2.0'
2:   info:
3:     description: |
4:       # Introduction
5:       This is the registry API for SwaggerHub. It allows you to access, manage, and update your APIs and Domains in SwaggerHub bypassing the Web application.
6:
7:       # Authentication
8:       Use your personal API Key: you may find it by visiting the [API Key page](https://app.swaggerhub.com/settings/apiKey).
9:
10:      version: 1.0.45
11:      title: SwaggerHub Registry API
12:      contact:
13:        name: SwaggerHub
14:        url: 'http://swaggerhub.com'
15:        email: info@swaggerhub.com
16:      host: api.swaggerhub.com
17:      tags:
18:        - name: APIs
19:          description: Operations for APIs
20:        - name: Domains
21:          description: Operations for Domains
22:      schemes:
23:        - https
24:      produces:
25:        - application/json
26:      paths:
27:        /specs:
28:          get:
29:            tags:
30:              - APIs
31:              - Domains
32:            summary: >-
33:              Retrieves a list of currently defined APIs and Domains in APIs.json
34:            format:
35:            descriptions: ''
36:            operationId: searchApisAndDomains
37:            parameters:
38:              - name: specType
39:                in: query
40:                description: |
41:                  Type of Swagger users to search
```

Last Saved: 08:27:47 am May 26, 2017 ✓ VALID Published

APIs: Operations for APIs

Method	Path	Description	Status
GET	/specs	Retrieves a list of currently defined APIs and Domains in APIs.json format.	Locked
GET	/apis	Retrieves a list of currently defined APIs in APIs.json format.	Locked
GET	/apis/{owner}	Retrieves an APIs.json listing of all APIs defined for this owner.	Locked
GET	/apis/{owner}/{api}	Retrieves an APIs.json listing for all API versions for this owner and API.	Locked
POST	/apis/{owner}/{api}	Saves the provided Swagger definition.	Unlocked
DELETE	/apis/{owner}/{api}	Deletes the specified API.	Locked
GET	/apis/{owner}/{api}/.collaboration	Gets API's collaboration.	Locked
PUT	/apis/{owner}/{api}/.collaboration	Updates API's collaboration.	Locked
DELETE	/apis/{owner}/{api}/.collaboration	Deletes API's collaboration.	Locked
GET	/apis/{owner}/{api}/{version}	Retrieves the Swagger definition for the specified API and version.	Locked
DELETE	/apis/{owner}/{api}/{version}	Deletes a particular version of the specified API.	Locked

Postman

The screenshot shows the Postman API documentation for the `POST /collections` endpoint. The left sidebar lists various collections, environments, mocks, monitors, and workspaces. The main content area is titled "POST Create Collection" and provides details about creating collections using the Postman Collection v2 format. It includes an example request in JavaScript and an example response in JSON.

Example Request:

```
var https = require('https');

var options = {
  'method': 'POST',
  'hostname': 'api.getpostman.com',
  'path': '/collections',
  'headers': {
    'X-Api-Key': '{{postman_api_key}}',
    'Content-Type': 'application/json'
  }
}
```

Example Response:

```
200 – OK
```

```
{
  "collection": {
    "id": "2412a72c-1d8e-491b-aced-93809c0e94e9",
    "name": "Sample Collection",
    "uid": "5852-2412a72c-1d8e-491b-aced-93809c0e94e9"
  }
}
```

Apiary

The screenshot displays the Apiary API Blueprint Editor interface, specifically the Notes API documentation. The left side shows the API Blueprint code in a monospaced font, detailing various endpoints like listing notes, creating a note, retrieving a note, and removing a note. The right side provides a detailed view of the API, including the introduction, reference, and a request section with a mock server and JavaScript code.

Notes API Blueprint Code:

```
1 FORMAT: 1A
2 HOST: http://api.google.com
3
4 # Notes API
5 Notes API is a *short texts saving* service similar to its physical paper presence on your table.
6
7 # Group Notes
8 Notes related resources of the **Notes API**.
9
10 ## Notes Collection [/notes]
11 ### List all Notes [GET]
12 + Response 200 (application/json)
13
14 [
15   {
16     "id": 1, "title": "Jogging in park"
17   },
18   {
19     "id": 2, "title": "Pick-up posters from post-office"
20   }
21 ]
22
23 + Request (application/json)
24
25   { "title": "Buy cheese and bread for breakfast." }
26
27 + Response 201 (application/json)
28
29   { "id": 3, "title": "Buy cheese and bread for breakfast." }
30
31 ## Note [/notes/{id}]
32 A single Note object with all its details.
33 + Parameters
34   + id (required, number, '1') ... Numeric 'id' of the Note to perform action on.
35 ## Retrieve a Note [GET]
36 + Response 200 (application/json)
37
38 + Header
39
40   X-My-Header: The Value
41
42 + Body
43
44   { "id": 2, "title": "Pick-up posters from post-office" }
45
46 ## Remove a Note [DELETE]
47 + Response 204
```

Notes API Documentation:

- INTRODUCTION:** Notes API is a short texts saving service similar to its physical paper presence on your table.
- REFERENCE:** Notes
- Notes Collection:**
 - List all Notes
 - Create a Note

Request Section:

Notes / Notes Collection / List all Notes

GET http://api.google.com/notes

Request

Mock Server ▾ JavaScript ▾ Try

```
1 var Request = new XMLHttpRequest();
2
3 Request.open('GET', 'http://private-10ca8-notesapi152.apiary-mock.com/notes');
4
5 Request.onreadystatechange = function () {
6   if (this.readyState === 4) {
7     console.log('Status:', this.status);
8     console.log('Headers:', this.getAllResponseHeaders());
9     console.log('Body:', this.responseText);
10  }
11 }
12
13 Request.send(JSON.stringify(body));
```

}

Ejemplo de Documentación de API





Swagger

TM



Entidades que utilizan OPENAPI

ebay

IBM

Q Rapid

Microsoft

SAP®

apiplatform.io

IFS

Geek.Zone

CISCO

Google

SWAGGER {

Es un estándar que describe, produce, consume y visualiza REST API, APIs web services.

```
cout << "Enter rows and columns for second matrix." << endl;
cin >> r2 >> c2;
cout << "Enter elements of first matrix." << endl;
for (int i = 0; i < r1; ++i)
    for (int j = 0; j < c1; ++j)
        cout << "Enter element a[" << i + 1 << j + 1 << "] : ";
        cin >> a[i][j];
cout << "Enter elements of matrix 1:" << endl;
for (int i = 0; i < r1; ++i)
    for (int j = 0; j < c1; ++j)
        cout << "Enter element b[" << i + 1 << j + 1 << "] : ";
        cin >> b[i][j];
cout << "Enter elements of second matrix." << endl;
for (int i = 0; i < r2; ++i)
    for (int j = 0; j < c2; ++j)
        cout << "Enter element c[" << i + 1 << j + 1 << "] : ";
        cin >> c[i][j];
cout << "Matrix A is" << endl;
for (int i = 0; i < r1; ++i)
    for (int j = 0; j < c1; ++j)
        cout << a[i][j] << " ";
    cout << endl;
cout << "Matrix B is" << endl;
for (int i = 0; i < r1; ++i)
    for (int j = 0; j < c1; ++j)
        cout << b[i][j] << " ";
    cout << endl;
cout << "Matrix C is" << endl;
for (int i = 0; i < r2; ++i)
    for (int j = 0; j < c2; ++j)
        cout << c[i][j] << " ";
    cout << endl;
```

PASOS{

Agregar
dependencia
en el pom.xml

```
<dependency>  
  <groupId>com.newrelic.agent.java</groupId>  
  <artifactId>newrelic-api</artifactId>  
  <version>4.2.0</version>  
</dependency>
```

PASOS{

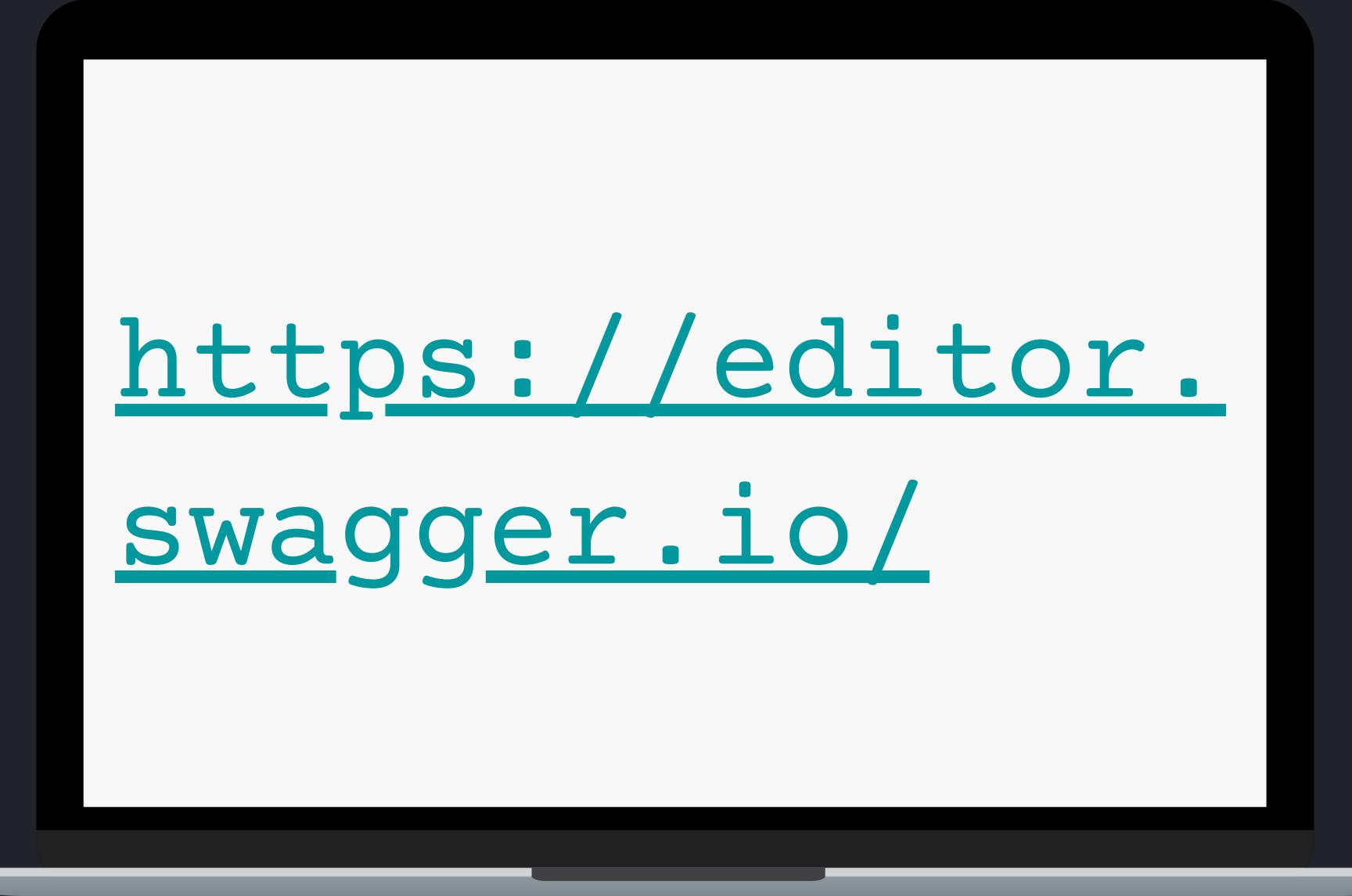
Agregarlos
import

```
import  
com.newrelic.api.  
agent.Trace
```

En el método
getrequest
agregamos la
URL de
nuestra API:

```
@Trace(metricName =  
"mobile/tigo/hn/digital/  
customer/credit/assessme  
nt/v1/consumer/{consumer  
}", dispatcher = true)
```

Vamos a
SwaggerEditor
e importamos
nuestro
archivo :



[https://editor.
swagger.io/](https://editor.swagger.io/)

}

Conclusiones {

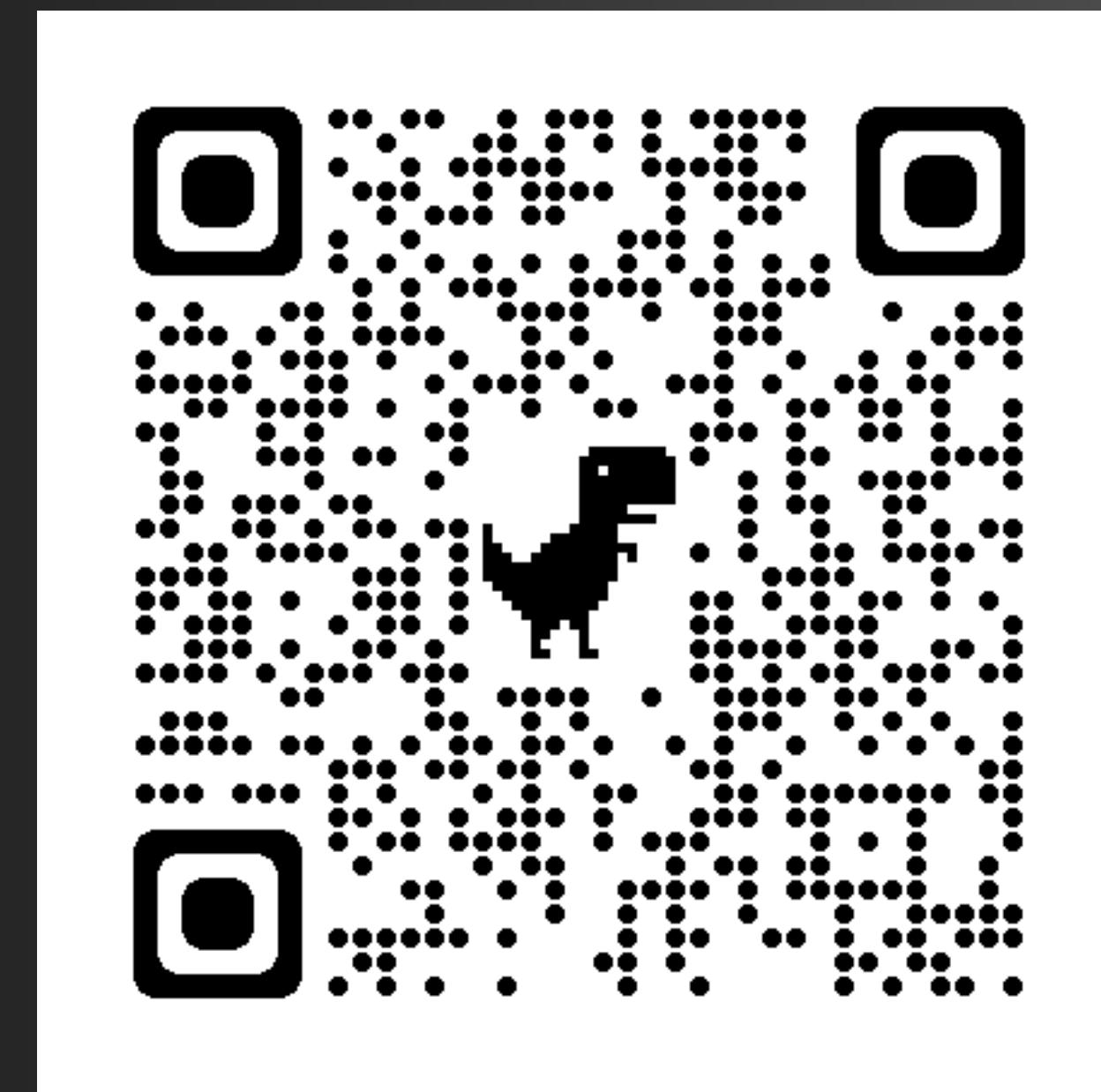
"La documentación es una carta de amor que le escribes a tu yo futuro".

--Damian Conway

}

Enlace de Presentacion

{



Documentación

}

```
<!--Grupo # 14 -->
```

Gracias {

<Por='tu atención'>

}